A MULTI-LEVEL INDEX STRUCTURE FOR VIDEO DATABASES

Lei Chen
School of Computer Science
University of Waterloo
l6chen@uwaterloo.ca

Vincent Oria
Department of Computer Science
New Jersey Institute of Technology
oria@cis.njit.edu

M. Tamer Özsu
School of Computer Science
University of Waterloo
tozsu@uwaterloo.ca

ABSTRACT

Several salient object-based data models have been proposed to model the video data, however, none of them proposed an index structure to handle the salient object-based queries efficiently. There are several indexing schemes that have been proposed for spatio-temporal relationships among objects and they are used to optimize *timestamp* and *interval* queries, which are rarely used in video database. Moreover, these index structures are designed without consideration of the granularity levels of constraints in salient objects and the characteristics of the video data. In this paper, we propose a multi-level index structure to efficiently handle the salient object-based quires with different levels of constraints. The characteristics of video data are also captured in the second level of the index structure designed to reduce the storage requirement.

1. INTRODUCTION

In recent years, content-based video retrieval has attracted increasing interest in many application fields, such as digital library, tele-conferences and surveillance system, etc. Based on the characteristics of video data, content-based video retrieval can be classified into three categories:

1. Retrieval based on visual feature [30, 9, 22, 29, 11, 16]: In this approach, a video is recursively broken down into *scenes*, *shots* and *frames*. Key frames are extracted from the shots and the scenes to summarize them, and visual features from the key frames are used to index the key frames.

2. Retrieval based on keyword or free text [23, 27, 12, 21, 13]: In this approach, a content description (annotation) layer is put on top of the video stream. Each descriptor can be associated with a logical video sequence or physically segmented shots or scenes.

3. Retrieval based on salient objects and their relationships (Salient objects are the physical objects that appear in the video data) [15, 18, 7, 6, 14, 3, 4]: In this approach, salient objects are extracted from the videos and the spatio-temporal relationships among them are described to express events or concepts.

Compared to the first and second approaches, the third approach is more intuitive and more suitable to the understanding of human beings, especially naive users. Users can directly manipulate salient objects, their properties, and the spatio-temporal relationships among them. Salient objects provide a better representation of video semantics. Basically, queries related to the spatio-temporal relationships of sailent objects can be classified into four types:

1. Salient object existence. In this type of query, users are only interested in the appearance of an object, for example, “Give me all the videos in which object a appears”.

2. Temporal relationships. These queries involve temporal relationships among objects in a video. For example, “Give me all the videos in which object a appears before object b”.

3. Spatial relationships. In these queries, users express simple directional or topological relationships among salient objects. For example, “Give me all the videos in which object a appears to the left of object b.”.

4. Spatio-temporal relationships. Users are concerned with the spatio-temporal relationships among salient objects in these queries. For example, “Give me all the videos in which object a appears on the left of object c before object b appears to the left of object c”.

In the four types of queries, users can specify conditions on spatial, temporal, or spatio-temporal relationships among the salient objects, and the queries return the *videos* that satisfy the conditions. Compared to *timestamp* queries and *interval* queries [28], which are two common types of queries for spatio-temporal databases, the above four types of queries are more intuitive for users of video databases. Timestamp queries retrieve all objects that intersect with a value range window at a specific time. Interval queries consider sequence of timestamps. The following are two examples of timestamp queries and interval queries, respectively [28]:

1. Search the objects within a rectangle R at time t_i;

2. Search the objects within a rectangle R from t_i to t_j.

Timestamp and interval queries are rarely used in video databases, since they require users to have a comprehensive knowledge of the video time line. It is very difficult for users to accurately specify the timestamp or time interval in which the events they are interested in occur. However they can be interested in finding the timestamps or intervals of interesting events.

A major problem in video databases is to find an effective index that can optimize video query processing. Sequential scan is quite inefficient for large video databases. The only index structure that has been designed for fast access to salient objects is the 3DR-tree [25], which indexes salient objects by treating the time as another dimension in the R-tree. However, simply treating the time as another dimension causes a lot of “dead” spaces [25]. Several index structures, such as the RT-tree [28], the HR-tree [19], and the MVR-tree [24], have been proposed for spatio-temporal databases. But these index structures are mainly designed to optimize *timestamp* and *interval* queries [28]. None of them is designed to optimize above salient object-based queries, and none of them captures the characteristics of salient objects. Furthermore, different users may have different granularity levels of constraints (e.g. salient objects, salient objects and spatial relationships, salient objects and spatio-temporal relationships). The existing index structures do not consider dealing with these issues.

In this paper, based on video data model [5], we propose a multi-level index structure to improve the efficiency of salient object-based queries with different levels of constraints. The key frames in which salient objects appear are indexed using an inverted file, which is the first level of our index structure. A variation of the HR-tree (VHR-tree) is designed to capture the characteristics of salient objects, which reduces the storage requirement compared to directly applying the HR-tree. The VHR-tree acts as the second level of the index structure. Each node of inverted file is linked to an element of the array which is used to index the timestamps in the VHR-tree.

The rest of the paper is organized as follows: Section 2 presents some related works on index structures. The video data model, base of our index structure is introduced in Section 3. Section 4 presents the multi-level index structure developed for salient objects. We conclude, in Section 5 and indicate some future directions.

2. RELATED WORK

As we mentioned earlier, not much has been done on indexing salient objects and their spatio-temporal relationships. The 3DR-tree [26] was originally proposed to speed up the operations of multimedia object composition and synchronization. The RT-tree [28], the HR-tree [19], and the MVR-tree [24] have been proposed to index spatio-temporal relationships in spatial and temporal database management systems (STDBMS). The RT-tree is a spatio-temporal version of the R-tree; it indexes all the spatio-temporal information in one R-tree, which makes it unmanageable when the number of objects that change is large. The MVR-tree can be considered as a variation of the 3DR-tree; it combines the concepts of mutli-version B-tree [2] and the 3DR-tree. In this section, we will mainly describe the HR-tree and the 3DR-tree.

2.1. HR-tree

Influenced by the idea of overlapping B-trees [17], Nascimento and Silva [19] proposed the HR-tree to reuse the common paths of R-trees at different timestamps. Figure 1 shows an example of HR-tree from timestamp T0 to T2. At timestamp T1, only object 3 changes to a new position 3a, therefore, the R-tree at T1 can reuse the subtrees B and C of R tree at T0 since they did not change at all. The same rules can be applied to R-tree at T2 at which only object 8 changes to a new position 8a. Compared to setting up a totally independent R-tree for each timestamp, HR-tree saves significant storage.

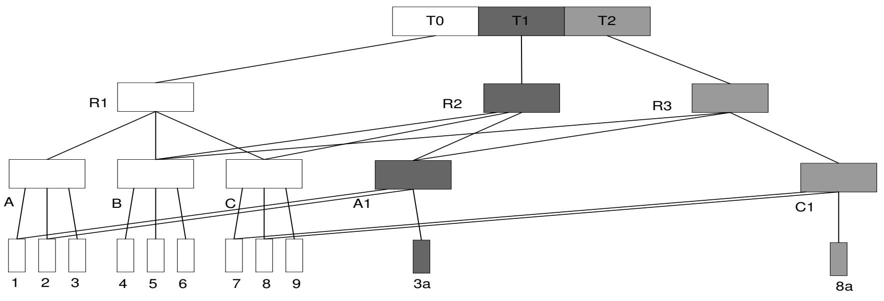

Fig. 1. An Example of HR-tree

The HR-tree can efficiently handle timestamp queries since such a query can be converted into a search in a static R-tree. However, for an interval query, all the trees whose timestamps are located inside the interval have to be searched.

2.2. 3DR-tree

3DR-tree [26] simply treats the time as another dimension and transforms a 2DR-tree to a 3DR-tree. It is designed to synchronize multimedia presentations and requires that indexed objects do not change their locations over time. Figure 2(a) shows four objects (A, B, C, D) in a 3D space, which are bound by two separated 3D Minimum Bounding Boxes R1 and R2, respectively. Figure 2(b) shows the corresponding 3DR-tree of those four objects.

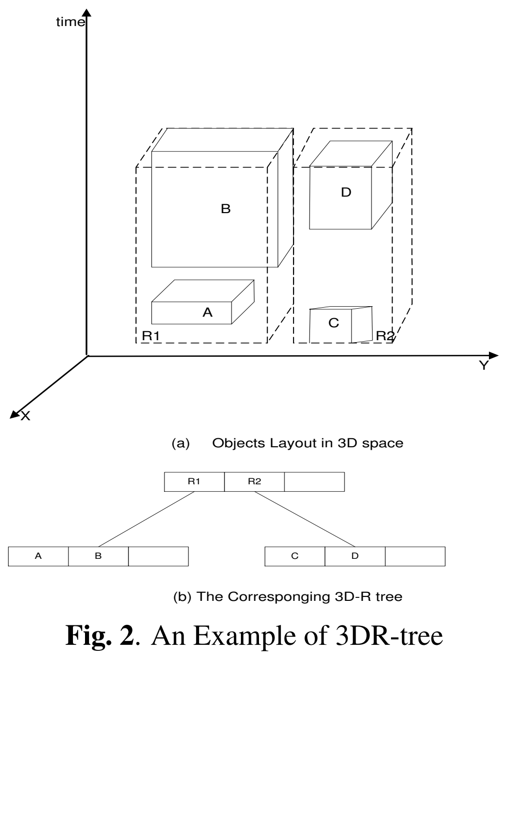
time
B
D
A
C
R1
R2
Y
X
(a) Objects Layout in 3D space
R1
R2
A
B
C
D
(b) The Corresponging 3D-R tree

Fig. 2. An Example of 3DR-tree

With the 3DR-tree, an interval query can be efficiently answered by converting it into a 3D object and finding the intersection with this object in the 3DR-tree. However, since long life span objects force the nodes that contain them to have long life span also, this introduces a lot of "dead" spaces and makes the timestamp query inefficient.

The aims of these index structures are to improve the efficiency of the system in dealing with the *timestamp* and *interval* queries. In order to answer queries that involve salient objects as we discussed in Section 1, all the timestamped R-trees have to be searched in the HR-tree or a large number of timestamp queries have to be executed against the 3DR-tree.

3. MODELING VIDEO DATA

In [4], we proposed a video data model, called common appearance interval (CAI) model, which represents videos in terms of salient objects and captures the appearance/disappearance of salient objects. The CAI model differentiates moving salient objects from static salient objects and different data structures are used to represent them respectively, which avoids saving redundant information of static salient objects. The trajectory of a moving object is represented as a dynamic attribute and a motion vector is created to act as the update function to derive the values of this attribute. Compared to the sampling method adopted by other models [6, 14], this method guarantees that no information about the moving objects is lost. We improved the CAI model by extending the DISIMA data model [20] to include video data [5]. The improved model captures both the spatio-temporal relationships among salient objects and the structure characteristics of video data.

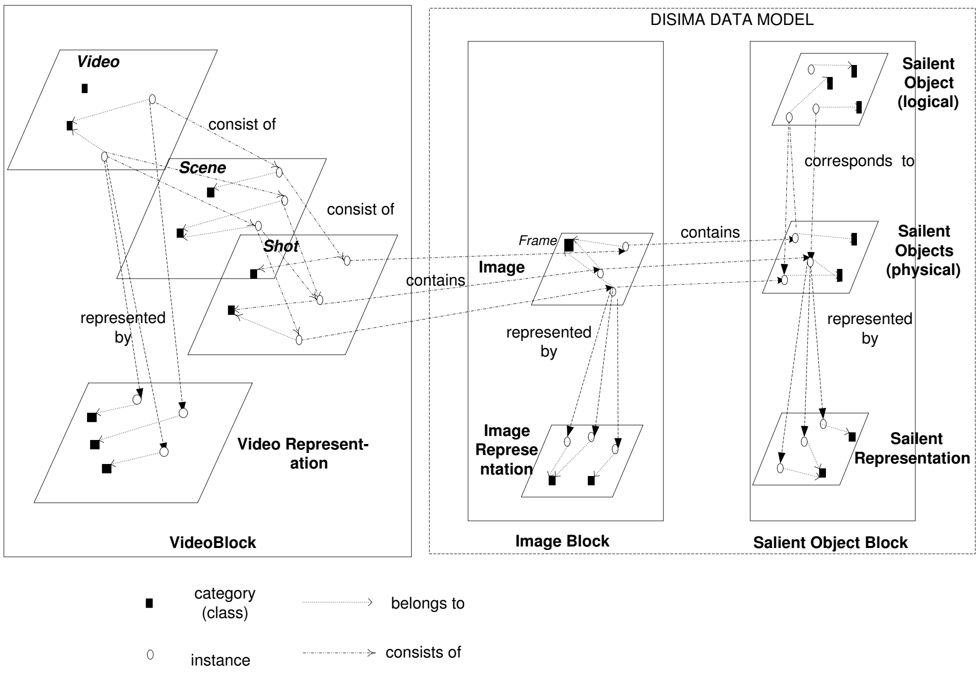

Fig. 3. Overview of the Video Data Model and its Links to the DISIMA Data Model

Figure 3 shows an overview of the improved video data model and its links to the DISIMA image data model. A video block is introduced to model video data. As defined in [20], a block represents a group of semantically related entities. In Figure 3, the video block has four layers: *video*, *scene*, *shot* and *video representation*. The relationships among the four layers are also shown. The basic composition unit of a video (scene, shot) is a video *frame*, which is treated as a special type of image. It inherits all the attributes from image entities and adds a new time attribute to model its temporal characteristics. In this data model, only the *key frames* are used to represent the contents of a shot. The relationship between key frames and shots sets up the connection between a video block and a DISIMA image block. The definitions of a key frame and its components are briefly summarized below.

Definition 1 A *key frame* is a video frame that is selected from a shot to represent the salient contents of the shot. A key frame KF_i is defined as a six-tuple

$$< i, R_i, C_i, D_i, SH_i, LS_i >$$

where

- i is the unique frame identifer;
- R_i is a set of representations of the raw frame (e.g. JPEG, GIF);
- C_i is the content of a key frame KF_i (see Definition 3);
- D_i is a set of descriptive alpha-numeric data associated with KF_i;
- SH_i is the shot (see Definition 4) to which KF_i belongs;
- LS_i is the *life-span* of the frame represented as a closed time interval $[T_s, T_e]$, which specifies the portion of the shot that KF_i represents. Since LS_i is within the shot, it must satisfy $LS_i \preccurlyeq SH_i.I_i$ where $\preccurlyeq$ is a "sub-interval" operation, defined as follows. Given two time intervals I_A and I_B, $I_A \preccurlyeq I_B$ if and only if $I_B.T_s \leq I_A.T_s$ and $I_A.T_e \leq I_B.T_e$, where T_s and T_e are the starting and end times of an interval.

In this data model, key frames are first selected through the automatic processes (using the on-the-shelf key frame selection algorithms [31, 29, 10, 8]) and manual interpretation processes are used to mark out the changes of salient objects. With these two steps, a key frame is selected to represent a duration within a shot in which the spatial relationships among salient objects contained in that video frame hold.

We identify, as in DISIMA, two kinds of salient objects: physical and logical.

Definition 2 A *physical salient object* is a part of a key frame and is characterized by a position (i.e. a set of coordinates) in the key frame space. A *logical salient object* is an abstraction of a set of physical salient objects and is used to give semantics to that set.

Based on the definitions of physical and logical salient objects, we define the content of a key frame.

Definition 3 C_i, the content of key frame KF_i, is defined by a pair

$$< \mathcal{P}^i, s >$$

where

- $\mathcal{P}^i$ is the set of physical salient objects which appear in KF_i and $\mathcal{P}$ is the set of all physical salient objects ($\mathcal{P} = \cup_i P^i$);
- $s : \mathcal{P}^i \rightarrow \mathcal{L}$ maps each physical salient object to a logical salient object, where $\mathcal{L}$ is the set of all logical salient objects.

Similar to images in DISIMA, two main representation models are used to represent key frames: the *raster* and the *vector*. Raster models are employed for image application and vector representations are used to reason about the spatial relationships among salient objects in a frame.

4. MULTI-LEVEL INDEXING OF SALIENT OBJECTS

Analysis of queries on salient objects (Section 1), we find that most video queries can be answered in two steps:

1. Find all the key frames that contain the specified video objects;

2. Check whether the spatio-temporal relationships of salient objects in the key frames satisfy the specified predicates of the query and return the videos (scenes, shots) that contain these key frames.

For the salient object existence level queries, only the first step is needed, while for the spatial, temporal and spatio-temporal relationship level queries, both steps are required. Therefore, we set up a two level index structure to satisfy different granularity levels of query constraints. The first level of the index structure is designed for quickly locating the key frames in which the specified salient objects appear. The second level of the index structure is set up for fast access spatio-temporal relationships among the salient objects.

4.1. First Level Index: Salient Object Inverted List

Since all the queries require finding the key frames that contain specific salient objects, quickly locating the key frames is quite important for answering these queries efficiently. We propose *Salient Object Inverted List* to index the key frames in which each salient object appears. The salient object inverted lists are the first level of the index structure.

Each salient object is linked to an inverted list which describes the key frames containing that salient object. A node of the inverted list contains the ID of a key frame and a pointer which points to the corresponding root of the second level index tree (Section 4.2). All the key frames in an inverted list are sorted according to the starting frames of their intervals. Figure 4 shows the structure of salient object inverted list.

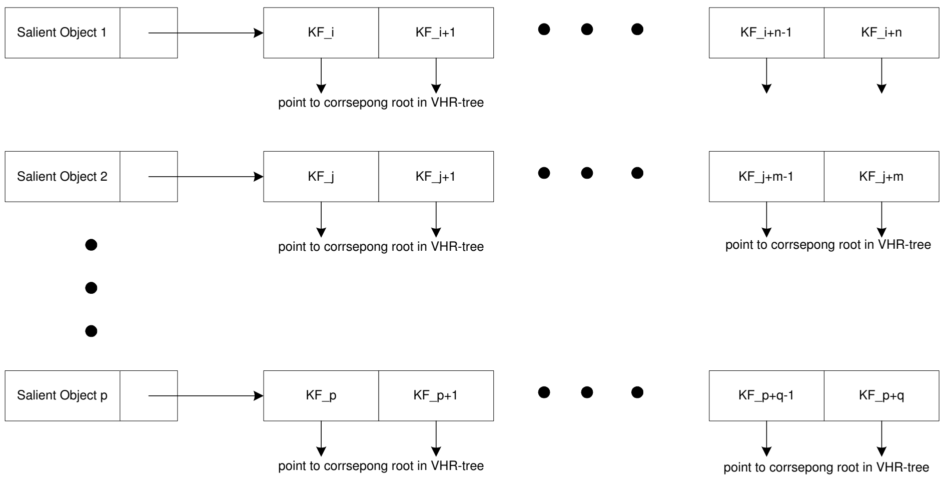
Salient Object 1
KF_i
KF_i+1
KF_i+n-1
KF_i+n
point to corrsepong root in VHR-tree
Salient Object 2
KF_j
KF_j+1
KF_j+m-1
KF_j+m
point to corrsepong root in VHR-tree
point to corrsepong root in VHR-tree
Salient Object p
KF_p
KF_p+1
KF_p+q-1
KF_p+q
point to corrsepong root in VHR-tree
point to corrsepong root in VHR-tree

Fig. 4. Structure of the salient object inverted list

Only using salient object inverted list, the following salient object existence and temporal relationship level queries can be answered:

- "Give me all the scenes in which salient object a appears"; Simply find the IDs of the salient objects in the inverted list, and return the related scenes that contain the key frames in the list. The time cost is bounded by the number of salient objects. Compared to sequential search, whose time cost is bounded by the number of key frames, using salient object inverted list in much faster since the number of key frames is much greater than the number of salient objects.

- "Give me all the scenes in which salient object a and b appears"; A multi-way join operation on salient object inverted lists will return the answer. The time cost will be bounded by the least number of key frames of the salient objects. In the worst case, when the objects appear in all the key frames, the time cost of using salient object inverted list will be same as that of the sequential search. However, it rarely happens that the salient objects appear in all of key frames of a large video database.

- "Give me all the scenes in which salient object a appears after b"; A multi-way join operation also can give the answer. In this paper, we define $after$ as immediately after, the same goes to $before$. Since $before$ and $after$ are "range" concepts, which are highly dependent on users' understanding, a range threshold δ can be set up for checking $before$ and $after$ relationship among key frames. The time cost analysis is same as the second example.

4.2. Second Level Index: VHR-tree

Queries on the spatial and spatio-temporal relationship levels contain constraints on spatial relationships among the salient objects. Therefore, an index structure on spatial relationships will be helpful. We select the HR-tree to index the spatial layout of the salient objects. The reason that we do not choose the 3DR-tree is that it does not consider reusing unchanged spatial information which results in space under utilization. However, we cannot use the HR-tree directly. In the video data, shot/reverse shot patterns are often used in dialog and action scenes, which are considered as "basic" sentences of a movie [1]. These shot/reverse shot patterns cause similar spatial layouts of salient objects to appear in an interleaving pattern. Based on our salient object-based video data model, the consecutive key frames may have totally different spatial relationships among salient objects or even have totally different groups of salient objects. If we apply the HR-tree directly, we have to create a new R-tree for each key frame. Hence, a sequence of similar independent R-trees will be created, which will waste a lot of storage space. Based on the definitions of key frames and the mechanisms that are used to select key frames, we design a variation of the HR-tree (called VHR-tree) to index the spatial relationships among the salient objects in key frames. These index trees are in turn used to construct the second level of our multi-level index structure.

When setting up the R-tree of a key frame, the two preceding key frames are checked in the VHR-tree (only the first is in the HR-tree). This ensures that two consecutive entries store totally different R-trees. Figure 5 shows how a VHR-tree handling the spatial patterns brought by shot/reverse shot. At timestamp T1, all the salient objects that appear at T0 disappear, and another nine new salient objects appear, therefore, another new R-tree is setup at T1. At timestamp T2, instead of only searching its immediate precedent R2 at T1 as the HR-trees does, the VHR-tree checks the R-trees at T1 and T0, and uses the unchanged part of the R-tree at T0. The same construction procedure applies for timestamp T3.

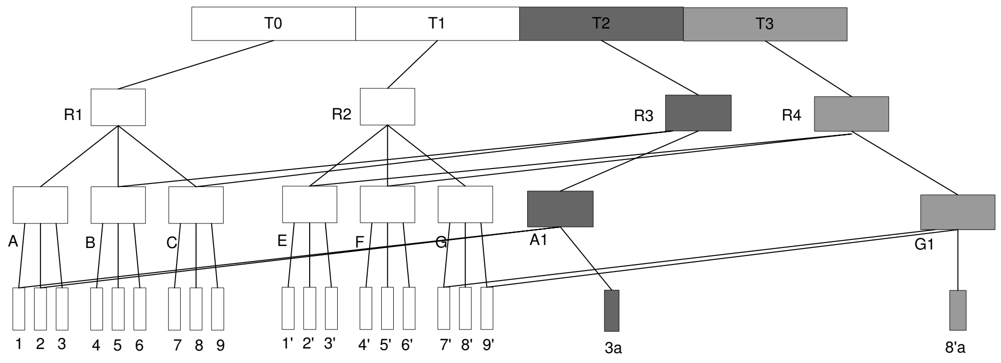

Fig. 5. The VHR-tree on a sequence of shots in shot/revershot pattern

With the first and second level indexes, we can answer spatial and spatio-temporal level queries quickly.

- "Give me all the scenes in which actor a appears to the left of actor b". First, through the first level index, we find the key frames that contain the salient objects a and b, and then trees (specified by key frames found in first step) in the VHR-tree will be searched to check the specified spatial relationship (left). Since only limited number of R-trees needs to be searched, compared to the sequential search on all the R-trees of VHR-tree, the cost in time of the multi-level index structure is normally reduced. In the worst case, when the salient objects appear in all the key frames, the time cost of using multi-level index structure and that of sequential search will be same.

- "Give me all the scenes in which actor a appears on the left of building c before actor b appears to the left of building c". First, we use the same steps as above to select the sequence of key frames which contain the spatial relationships (a left c). Then, for each discontinuous key frame in the selected sequence, its immediate next key frame is checked to see whether the spatial relationship (b left c) holds. Compared to the sequential search, using our multi-level index structure save the time on the first step based on the similar analysis as the above case.

4.2.1. VHR-tree Operations

In the VHR-tree, two preceding key frames are searched for sharable common branches when creating the index tree of current timestamp. Therefore, the $insert$ operation of original HR-tree has to change. Furthermore, in order to quickly locate the R-tree which may have the common path, we also modify the array which was originally used to store only the timestamps to store both timestamps of key frames and the IDs of the salient objects in those key frames. For the $delete$ operation, we adopt the original algorithm of the HR-tree for the VHR-tree.

When we create an index for a key frame in the VHR-tree, if the key frame and its immediate precedent key frame contain the same set of salient objects and only differ on the positions of salient objects, only the direct precedent key frame is needed to check the common branches of the trees. Therefore, the original $insert$ operation of the HR-tree can be used here. If the key frame and its direct precedent contain different salient objects, we have to check two preceding precedents in order to find as much as possible shareable index structure for the key frame. In this paper, the two consecutive precedents are selected for comparison. Among these two frames, the one that contains more common salient objects with respect of the key frame being indexed will be selected and a R-tree will be created by applying the original $insert$ operation on the corresponding R-tree of the selected key frame.

From the above discussion, it is evident that except the operation which is used to find the number of common salient objects between key frames, we do not introduce any other overhead on the $insert$ operation of the VHR-tree compared to the HR-tree and we only apply $insert$ operation on the selected R-tree of the key frame. By using the merge join, the operation of finding the number of common salient objects costs $O(n)$, where n is the largest number of salient objects in key frames.

5. CONCLUSION AND FUTURE WORK

Querying video databases on salient objects is quite different from querying spatio-temporal databases. The two fundamental types of queries in spatio-temporal databases, *timestamp* and *interval* queries, are rarely used in salient object-based video diabases, since users normally do not have any knowledge about the timestamps or intervals in which some specified event happen. What they are interested in is exactly to retrieve those timestamps or intervals! Therefore, the index structures of spatio-temporal databases cannot be directly applied to salient object-based databases. Another fact about salient object-based queries, ignored by earlier proposed indexing techniques, is that users may have different levels of constraints when they posed queries against the video database. Single level index structure may not be efficient for queries with different level constraints.

In this paper, we proposed a multi-level index structure based on our salient object-based data model, it has following advantages over other indexing schemes:

1. It designed with the consideration that different users may pose different level constraints on their queries.

2. It designed to optimize salient object-based queries, posed against a video database with no knowledge about the video time line.

3. The second level of our index structure, the VHR-tree, extends the original HR-tree in order to capture the characteristics of video data, which reduces the storage requirement of the index structure.

In our future work, we will design some experiments to test the efficiency of using index structure compared to that of sequential search and using the 3DR-tree. Furthermore, possible index structures for fast retrieval of trajectories of moving objects will be studied.

6. REFERENCES

[1] D. Arijon, *Grammar of the Film Language*, Focal Press, 1976.

[2] B. Becker, S. Gschwind, T. Ohler, and B. Seeger, "An asymptotically optimal multi-version B-tree," *VLDB Journal*, vol. 5, no. 4, pp. 264–275, 1996.

[3] S-C. Chen and R. L. Kashyap, "A spatiotemporal semantic model for multimedia presentations and multimedia database systems," *IEEE Transaction on Knowledge and Data Engineering*, vol. 13, no. 4, pp. 607–622, 2001.

[4] L. Chen and M. T. Özsu, "Modeling video objects in video databases," in *Proceedings of IEEE International Conference on Multimedia and Expro*, pp. 217–221, August 2002.

[5] L. Chen, M. T. Özsu and V. Oria, "Modeling video data for content-based queries: extending the DISIMA image data model," in *Proceedings of International Conference on Multi-Media Modeling*, January 2003, submitted.

[6] Y.F. Day, S. Dagtas, M. Iino, A. Khokhar, and A. Ghafoor, "Object-oriented conceptual modeling of video data," in *Proceedings of the Eleventh International Conference on on Data Engineering*, pp. 401–408, March 1995.

[7] A. Del Bimbo, E. Vicario, and D. Zingoni, "Symbolic description and visual querying of image sequences using spatio-temporal logic," *IEEE Transaction on Knowledge and Data Engineering*, vol. 7, no. 4, pp. 609–622, 1995.

[8] F. Dufaux, "Key frame selection to represent a video," in *Proceedings of IEEE International Conference on Image Processing*, pp. 275–278, Jan 2000.

[9] B. Günsel, A.M. Ferman, and A. M. Tekapl, "Temporal video segmentation using unsupervised clustering and semantic object traking," *Journal of Electronic Imaging* , vol. 7, no. 3, pp. 592-604, 1998.

[10] B. Günsel and A. M. Tekapl, "Content-based video abstraction," in *IEEE Proceedings of International Conference on Image Processing*, pp. 128–131, 1998.

[11] A. Hanjalic, R.L. Lagendijk, and J.Biemond, "Automatically segmenting movies into logical story units," in *Proceedings of International Conference on Visual Information Systems*, pp. 229–236, 1999.

[12] R. Hjelsvold and R. Midtstraum, "Modelling and querying video data," in *Proceedings of 20th International Conference on Very Large Data Bases*, Santiago, Chile, pp. 686–694, 1994.

[13] H.T. Jiang, D. Montesi, and A. K. Elmagarmid, "Videotext database systems," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pp. 344–351, June 1997.

[14] W. Al-Khatib and A. Ghafoor, "An approach for video meta-data modeling and query processing," in *Proceedings of ACM Multimedia*, pp. 215–224, 1999.

[15] J.Z. Li, M.T. Özsu, and D. Szafron, "Modeling of moving objects in a video databas," *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pp. 336–343, 1997.

[16] W. Mahdi, M. Ardebilian, and L.M.Chen, "Automatic video scene segmentation based on spatial-temporal clues and rhythm," *Networking and Information Systems Journal*, vol. 2, no. 5, pp. 1–25, 2000.

[17] Y. Manolopoulos and G. Kapetanakis, "Overlapping B^+-tree for temporal data," in *Proceedings of the 6th JCIT conference*, pp. 491–498, 1990, .

[18] M. Nabil, A. H. H. Ngu, and J. Shepherd, "Modeling moving objects in multimedia database," in *Proc. of the 5th Conf. on Database Systems for Advanced Applications*, Melbourne, Australia, pp. 67–76, 1997.

[19] M.A. Nascimento and J.R.O. Silva, "Towards historical R-trees," in *Proceedings of ACM Symposium on Applied Computing (ACM-SAC)*, pp. 235–240, 1998.

[20] V. Oria, M.T. Ozsu, L. Liu, X. Li, J.Z. Li, Y. Niu, and P. Iglinski, "Modeling images for content-based queries: The disima approach," in *Proceedings of Second International Conference on Visual Information Systems*, pp. 339–346, Dec 1997.

[21] E. Oomoto and K. Tanaka, "OVID: Design and implementation of a video-object database system," *IEEE Transaction on Knowledge and Data Engineering*, vol. 4, no. 5, pp. 629–643, 1993.

[22] Y. Rui, T. S. Huang, and S. Mehrotra, "Exploring video structure beyond the shots," in *Proceedings of IEEE International Conference on Multimedia Computing and Systems*, pp. 237–240, 1992.

[23] T.G.A. Smith and G.Davenport, "The stratification system: A design environment for random access video," in *Proceedings of Workshop on Networking and Operating Syste Support for Digitial Audio and Video*, 1992.

[24] Y.F. Tao and D. Papadias, "MV3R-tree: A spatio-temporal access method for timestamp and interval queries," in *Proceedings of 27th International Conference on Very Large Data Bases*, Rome, Italy, pp. 431–440, 2001.

[25] Y. Theodoridis, T. Sellis, A. N. Papadopoulos, and Y. Manolopoulos, "Specifications for efficient indexing in spatiotemporal databases," in *Proceedings of IEEE International Confernece on SSDBM*, pp. 242–245, 1998.

[26] M. Vazirgiannis, Y. Theodoridis, and T. Sellis, "Spatio-temporal composition and indexing for large multimedia applications," *Multimedia Systems*, vol. 6, pp. 284–298, 1998.

[27] R. Weiss, A. Duda, and D.K Gifford, "Composition and search with a video algebra," *IEEE Multimedia*, pp. 12–25, 1994.

[28] X. Xu, J. Han, and W. Lu, "RT-tree: An improved R-tree index structure for spatiotemporal databases," in *Proceedings of 4th International Symposium on Spatial Data Handling (SDH)*, pp. 1040–1049, 1990.

[29] M.M. Yeung and Boon-Lock Yeo, "Time-constrained clustering for segmentation of video into story units," in *Proceedings of 13th International Conference on Pattern Recognition*, vol. 3, pp. 375–380, 1996.

[30] H.J. Zhang, A. Kankanhalli, and S.W. Smoliar, "Automatic partitioning of full-motion video," *Multimedia Systems*, vol. 1, pp. 10–28, 1993.

[31] H.J. Zhang, C.Y. Low, S.W. Smoliar, and J.H.Wu, "Video parsing, retrieval and browsing: An integrated and content based solution," in *Proceedings of ACM Multimedia*, San Francisco, CA, pp. 15–24, 1995.